Teaching Statement

Calvin Newport

November 5, 2010

Earlier this fall, I received an e-mail from a student whom I'll call David, a computer science major from Cornell. He was overwhelmed by a course load that required five weekly problem sets. "How do you propose I balance all these classes and still strive to score well?," he asked, before adding, with resignation: "schedules like mine are impossible to avoid."

I had heard this story before, and I wasn't buying it.

"Students often report that they have an impossible semester, and when I tell them to do less, they claim that it's impossible," I began my reply. "Then I push back some more, and they eventually admit that perhaps *some* reductions might be possible. I hope this is true for you." I gave David simple advice, *do less, but do what you do better,* and then outlined some strategies for achieving the latter goal.

Not long ago, I heard back from David. He admitted that perhaps it was not strictly necessary that he take such a hard schedule. He had reduced his load and since found room to actually learn the material—not just survive. "I realized that even though there's this notion that engineers must always have a very difficult schedule...it's simply not true," he said. He had taken control of his student life, and I do not doubt he will continue to reap the rewards of this decision throughout the remainder of his time at Cornell.

David's e-mail is one of the more than 100 such messages I receive in a typical month. These students write me in my capacity as a national authority on student advice. During the past seven years, I wrote three popular student advice guides (published by Random House) [13, 14, 15], and started the web's most trafficked student advice blog, *Study Hacks* [11], which receives over $250,000$ visits a month. I have appeared as a student expert on all three major broadcast networks and dozens of radio programs. I give talks on the subject around the country, including at top academic institutions such as Harvard, Dartmouth, and MIT. Students at dozens of colleges around the world have followed my *Study Hacks on Campus* curriculum in self-organized clubs [12], my advice is quoted in major publications (e.g., [8]), taught on college campuses (e.g., [1, 2]), and has been included in college textbooks (e.g., [5]).

To be clear, my work as a student advisor is a side pursuit. It augments my traditional experience in teaching and advising: I was a teaching assistant for both a theory course (*Distributed Algorithms*, taught by Nancy Lynch) and a systems course (*Network and Computer Security*, taught by Ron Rivest) at MIT, a guest lecturer for MIT's *Distributed Algorithms for Mobile Ad Hoc Networks* graduate seminar, and co-advisor to five undergraduate research students (through MIT's UROP program), and two masters of engineering students. This side pursuit, however, is important, as it influences my approach to teaching in a crucial, but often overlooked, manner: I believe that teaching students how to *be effective computer scientists*—taking control of their student careers, and avoiding the burn out that follows blind overwork—is as important as teaching them the content of computer science. With this in mind, I structured this statement in three brief sections. The first two sections are traditional: my teaching plans and philosophy on teaching computer science material. The third is unique to my extensive experience as a student advisor: my philosophy on teaching students the strategic side of success in higher education.

Teaching Plans. My research and teaching experience includes both sides of the theory/systems split. This allows me to step comfortably into teaching both the traditional theory courses, such as algorithms, theory of computation, and distributed algorithms, as well as the more systems-oriented networking courses. In addition, as outlined in my research statement, I believe strongly in combining theory and practice in the networking field. Accordingly, I am interested in developing a new course that reflects this concept. I propose designing a distributed algorithms course that grounds cutting-edge theory results in the field (e.g., the recent emphasis on wireless/mobile networks and multicore systems) in an understanding of the relevant systems work. A student learning recent theory results on structuring multihop wireless networks (e.g., [7]), for example, should also be exposed to the reality of these networks, such as the pioneering work on MIT's RoofNet project [3]. In my vision, students would as comfortable with Bar-Yehuda et al.'s canonical *Decay* algorithm [4] as they are with the ETX link quality metric [3]. This course could be designed for both the undergraduate or graduate levels.

Teaching Philosophy Part 1: Teaching Computer Science. A defining moment in my computer science education was Prasad Jayanti's Introduction to Theoretical Computer Science course, taught at Dartmouth College. Jayanti is obsessed with insight. If his students could not grasp an idea in their gut, he was not happy. I remember him dancing across the front of a crowded classroom in Dartmouth's Carson Hall, killing off non-determinsitic incarnations of himself as they failed to find a polynomial-time solution. "Kill this Prasad with my sword!", he would cry out. No one in Jayanti's class had difficulty understanding the difference between P and NP.

Students are intimidated by the mathematical details in technical fields such as theoretical computer science. For many, proofs seem unapproachable—intricate puzzles that fall fully-formed from the minds of other people; *math people*. Insight-centric teaching counters this myth. My approach is to start with the main idea and why we think it might work. The rest—the mathematics, the induction—are simply a convenient language for expressing these insights in a precise way. It may be a pain to learn to speak these languages well, but this should not be a source of intimidation. The real challenge (and fun) are the insights that precede this expression. This is a challenge that most, even those intimidated by mathematics, feel that they can achieve. In my experience as a teaching assistant and academic advisor to students at MIT, I have begun practicing an insight-centric pedagogy. (I have also been developing the concepts in my informal role as a student advice expert: an article I wrote on the subject, for example, has been picked up by numerous other web sites [10].) Certainly, however, this is an ability that can only grow with classroom experience. I look forward to this growth.

Teaching Philosophy Part 2: Teaching How to Be a Successful Computer Scientist. As mentioned, to me, an exciting prospect of becoming a professor is developing a collection of pedagogical techniques to help students take control of their student lives—finding success on their own terms, and avoiding the stress and anxiety that follows blind overwork. Such issues are usually left to campus counselors and academic skills departments, but I think it should be addressed directly in the classroom. Of course, I recognize that this process must be approached with patience and humility, and I plan to let the aggressiveness of my strategies grow only with my teaching experience. That being said, I want to describe three preliminary ideas I am considering to reinforce this philosophy.

Staggered Problem Sets: In addition to the standard problem set deadlines, students can opt to submit individual problems early. The earlier a problem is submitted, the more extra credit points the student receives. From my experience, if you can induce a student to abandon a *night before* mindset for even just a small number of projects, there is a strong chance that he or she might drop the procrastination habit altogether.

Strategy Memos: After each exam, I ask the top ten performers to share with me their approach to studying. I then send to the class a description of the most interesting of these strategies (anoynomized, of course). Compelling research from Columbia University [6], supported by my own advising experiences [9], shows that in technical classes, students who adopt an *adaptive mindset*—i.e., believe that smart strategies matter more than fixed intelligence—perform better better and are happier. My strategy memos are designed to induce this mindset: It is not just about *smart kids* versus *dumb kids*—treating the preparation process with some respect and creativity, and thinking through your review strategy carefully, matters more than exaggerated notions of innate ability.

Mandatory Shop Talk: After the first midterm, each student must sign up for a 20 minute slot to sit down with me and talk shop about how they are approaching the classroom demands, what is working, what is not, and what they might do to make things better. Like David, whose story opened this statement, sometimes all a burnt out student needs is someone with some authority to look him or her in the eye, and say with conviction: *"You don't have to live this way. Here's an alternative."* I have helped thousands, in person and over e-mail, and many tens of thousands through my books, take control of their student lives. The students I teach should receive no less than this same attention.

References

[1] Dartmouth College Academic Skills Center. Learning Links. `http://www.dartmouth.edu/~acskills/cal-newport.html`.

[2] William & Mary College: Before the First Day of Class. `http://tinyurl.com/2eoj8gz`.

[3] D. Aguayo, J. Bicket, S. Biswas, R. Morris, B. Chambers, and D. De Couto. MIT Roofnet. In *Proceedings of the International Conference on Mobile Computing and Networking*, 2003.

[4] R. Bar-Yehuda, O. Goldreich, and A. Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104–126, 1992.

[5] V. Gordon and T. Minnick. *Foundations: A Reader for New College Students, 5th Edition*. CENGAGE Learning, 2011.

[6] H. Grant and C. Dweck. Clarifying Achievement Goals and Their Impact. *Journal of Personality and Social Psychology*, 85(3), 2003.

[7] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Fast deterministic distributed maximal independent set computation on growth-bounded graphs. In *DISC*, 2005.

[8] J. Mattews. More Than One Way to Make the Grade. (The Washington Post Education Section). `http://www.washingtonpost.com/wp-dyn/content/article/2006/12/11/AR2006121100908.html`.

[9] C. Newport. Help! I Got a C on my Orgo Exam! What Should I Do? (Study Hacks Blog). `http://calnewport.com/blog/2010/04/01/i-got-a-c-on-my-orgo-exam-what-should-i-do/`.

[10] C. Newport. How to Ace Calculus (Study Hacks Blog). `http://calnewport.com/blog/2008/11/14/how-to-ace-calculus-the-art-of-doing-well-in-technical-c`

[11] C. Newport. Study Hacks. `http://www.calnewport.com/blog`.

[12] C. Newport. The Study Hacks on Campus Program. `http://www.calnewport.com/shoc/`.

[13] C. Newport. *How to Win at College*. Broadway/Random House, 2005.

[14] C. Newport. *How to Become a Straight-A Student*. Broadway/Random House, 2006.

[15] C. Newport. *How to Be a High School Superstar*. Broadway/Random House, 2010.